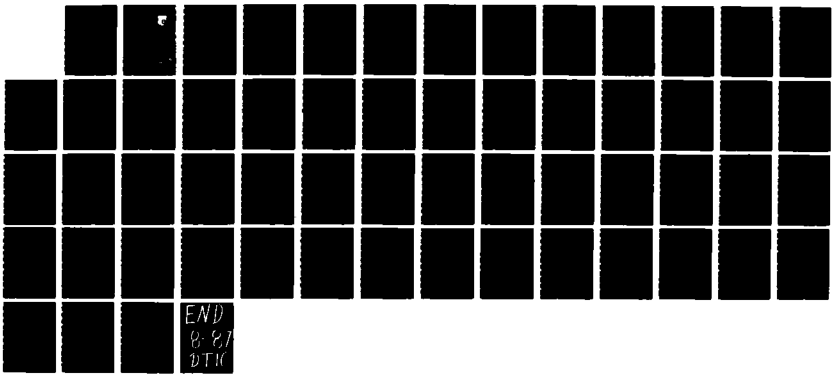
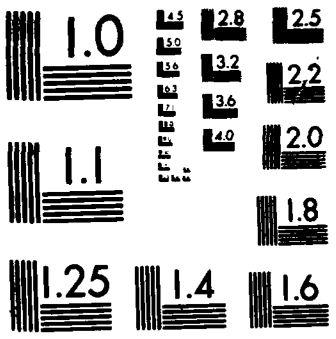
AD-A182 585 INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) VOLUME 8 USER INTERFACE SUBS (U) GENERAL ELECTRIC CO SCHENECTADY NY PRODUCTION RESOURCES CONSU 1/1

UNCLASSIFIED V CROSS ET AL 01 NOV 85 DS-620144402B F/G 12/5 NL



END
8-87
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

2

AD-A182 585

AFWAL-TR-86-4006
Volume VIII
Part 17

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 17 - Forms Driven Form Editor Development Specification

General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345

Final Report for Period 22 September 1980 - 31 July 1985

November 1985

MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6533

NOTICE

This technical report has been reviewed and is approved for publication.

David L. Judson

DAVID L. JUDSON, PROJECT MANAGER
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

5 Aug 1986
DATE

FOR THE COMMANDER:

Gerald C. Shumaker

GERALD C. SHUMAKER, BRANCH CHIEF
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

7 Aug 86
DATE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

A182 585

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S): AFWAL-TR-86-4006 Vol VIII, Part 17
6a. NAME OF PERFORMING ORGANIZATION: General Electric Company Production Resources Consulting; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: AFWAL/MLTC
6c. ADDRESS (City, State and ZIP Code): 1 River Road Schenectady, NY 12345	7b. ADDRESS (City, State and ZIP Code): WPAFB, OH 45433-6533
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Materials Laboratory Air Force Systems Command, USAF; 8b. OFFICE SYMBOL (If applicable): AFWAL/MLTC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F33615-80-C-5155
8c. ADDRESS (City, State and ZIP Code): Wright-Patterson AFB, Ohio 45433	10. SOURCE OF FUNDING NOS.
11. TITLE (Include Security Classification): (See Reverse)	

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
78011F	7500	62	01

12. PERSONAL AUTHOR(S): Cross, Valerie and Morenc, Carol and Roby, Penny

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Final Technical Report	22 Sept 1980 - 31 July 1985	1985 November	76

16. SUPPLEMENTARY NOTATION: ICAM Project Priority 6201. The computer software contained herein are theoretical and/or references that in no way reflect Air Force-owned or -developed computer software.

17. COSATI CODES

FIELD	GROUP	SUB. GR.
1308	0905	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This specification establishes the development, test and qualification requirements of a computer program identified as the Forms Driven Form Editor (FDFE) and the design of the FDFE. The FDFE is a software tool for creating and initializing form definitions. The FDFE displays a series of screens which request information from the user and visually show the form under construction. Once a form has been completed, the FDFE stores the form definition constructs needed to recreate the form. The stored form can be selected and modified.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: David L. Judson	22b. TELEPHONE NUMBER (Include Area Code): 513-255-6976; 22c. OFFICE SYMBOL: AFWAL/MLTC

DD FORM 1473, 83 APR   EDITION OF 1 JAN 73 IS OBSOLETE.   Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

11. Title

Integrated Information Support System (IISS)
Vol VIII - User Interface Subsystem
Part 17 - Forms Driven from Editor Development
Specification

A S D 86 0034
9 Jan 1986

DS 620144402B
1 November 1985

PREFACE

This product specification covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Alan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

TASK 4.2

Subcontractors	Role
Boeing Military Aircraft Company (BMAC)	Reviewer.
D. Appleton Company (DACOM)	Responsible for IDEF support, state-of-the-art literature search.
General Dynamics/ Ft. Worth	Responsible for factory view function and information models.

Subcontractors	Role
Illinois Institute of Technology	Responsible for factory view function research (IITRI) and information models of small and medium-size business.
North American Rockwell	Reviewer.
Northrop Corporation	Responsible for factory view function and information models.
Pritsker and Associates	Responsible for IDEF2 support.
SofTech	Responsible for IDEFO support.

TASKS 4.3 - 4.9 (TEST BED)

Subcontractors	Role
Boeing Military Aircraft Company (BMAC)	Responsible for consultation on applications of the technology and on IBM computer technology.
Computer Technology Associates (CTA)	Assisted in the areas of communications systems, system design and integration methodology, and design of the Network Transaction Manager.
Control Data Corporation (CDC)	Responsible for the Common Data Model (CDM) implementation and part of the CDM design (shared with DACOM).
D. Appleton Company (DACOM)	Responsible for the overall CDM Subsystem design integration and test plan, as well as part of the design of the CDM (shared with CDC). DACOM also developed the Integration Methodology and did the schema mappings for the Application Subsystems.

Subcontractors	Role
Digital Equipment Corporation (DEC)	Consulting and support of the performance testing and on DEC software and computer systems operation.
McDonnell Douglas Automation Company (McAuto)	Responsible for the support and enhancements to the Network Transaction Manager Subsystem during 1984/1985 period.
On-Line Software International (OSI)	Responsible for programming the Communications Subsystem on the IBM and for consulting on the IBM.
Rath and Strong Systems Products (RSSP) (In 1985 became McCormack & Dodge)	Responsible for assistance in the implementation and use of the MRP II package (PIOS) that they supplied.
SofTech, Inc.	Responsible for the design and implementation of the Network Transaction Manager (NTM) in 1981/1984 period.
Software Performance Engineering (SPE)	Responsible for directing the work on performance evaluation and analysis.
Structural Dynamics Research Corporation (SDRC)	Responsible for the User Interface and Virtual Terminal Interface Subsystems.

Other prime contractors under other projects who have contributed to Test Bed Technology, their contributing activities and responsible projects are as follows:

Contractors	ICAM Project	Contributing Activities
Boeing Military Aircraft Company (BMAC)	1701, 2201, 2202	Enhancements for IBM node use. Technology Transfer to Integrated Sheet Metal Center (ISMC).

Contractors	ICAM Project	Contributing Activities
Control Data Corporation (CDC)	1502, 1701	IISS enhancements to Common Data Model Processor (CDMP).
D. Appleton Company (DACOM)	1502	IISS enhancements to Integration Methodology.
General Electric	1502	Operation of the Test Bed and communications equipment.
Hughes Aircraft Company (HAC)	1701	Test Bed enhancements.
Structural Dynamics Research Corporation (SDRC)	1502, 1701, 1703	IISS enhancements to User Interface/Virtual Terminal Interface (UI/VTI).
Systran	1502	Test Bed enhancements. Operation of Test Bed.

TABLE OF CONTENTS

APPENDICES

FIGURES

SECTION 1

SCOPE

1.1 Identification

This specification establishes the development, test and qualification requirements of a computer program identified as the Forms Driven Form Editor (FDFE) and the design of the FDFE. The FDFE is one configuration item of the Integrated Information Support System (IISS) User Interface.

1.2 Functional Summary

The Forms Driven Form Editor is a software tool for creating and initializing form definitions. The FDFE displays a series of screens which request information from the user and visually show the form under construction. Once a form has been completed, the FDFE stores the form definition constructs needed to recreate the form. The stored form can be selected and modified.

The FDFE functions can be viewed on three different levels:

A. Conceptual

The user views the FDFE as a tool which can perform the following functions on form language source:

1) insert, select, modify, and drop form language source
2) insert, select, modify, and drop forms within a forms language source
3) select form language source and combine sections of one or more form language sources into one form language source
4) list all the form language sources
5) rename and copy form language source
6) list all forms created in a form language source

The FDFE may be used by the editor to perform the following operations on the compiled form definitions:

1) drop the compiled form definition
2) list all the compiled form definitions
3) view the compiled form definition

The fields of a form may be operated on in the following ways:

1) insert, drop, modify, select and list fields
2) copy fields from one form to another form
3) for these operations, all the different edit modes may be used

B. External

The runtime UI or UIMS views the FDFE, which is part of the UIDS, as an application program which uses the form processor. Data to be selected or stored comes from or is passed to the Common Data Model (CDM) in the integrated implementation; otherwise, a file system is used. The FDFE also interacts with the Forms Language Compiler (FLAN) to translate between the forms language source and the compiled form definition.

C. Internal

The FDFE is a C program which makes extensive use of form language sources and compiled forms, performs interactive user input/output via the Form Processor (FP), and uses the FP to manage the compiled forms. The internal form data structure is the same as that used by the Form Processor.

Since editors require extensive testing and revision before achieving true functionality, changes to the presented design may occur through use of the FDFE to design and develop forms for application.

SECTION 2

DOCUMENTS

2.1 Reference Documents

A small amount of research has focused on editor design; almost none has examined forms driven form editors. Some tangential references are listed below:

[1] "Designing a Portable Natural Language Database Query System", S. J. Kaplan, ACM Trans. on Database Sys. 9(1), 1984.

[2] "Document Formatting System: Survey, Concepts and Issues", R. Furuta, J. Scofield, A. Shaw, ACM Comp. Surveys 14(3), 1982.

[3] "Formal Grammar and Human Factors Design of an Interactive Graphics System", P. Reisner, IEEE Trans. on Software Eng. 7(2), 1981.

[4] HUMAN PERFORMANCE ENGINEERING: A GUIDE FOR SYSTEM DESIGNERS, R. Bailey; Prentice-Hall, Inc., (1982).

[5] ICAM DOCUMENTATION STANDARDS, ICAM DOCUMENT IDS 150120000C, 15 September 1983.

[6] "Interactive Editing Systems: Parts I and II", N. Meyrowitz and Andries van Dam, ACM Comp. Surveys 14(3), 1982.

[7] Structural Dynamics Research Corporation, IISS Form Editor User Manual, UM 620144400B, 1 November 1985.

[8] Structural Dynamics Research Corporation, IISS Form Processor User Manual, UM 620144200B, November 1, 1985.

[9] Structural Dynamics Research Corporation, IISS Terminal Operator Guide, OM 620144000 , November 1, 1985.

[10] Structural Dynamics Research Corporation, Report Writer Development Specification, DS 620144501, November 1, 1985.

[11] Structural Dynamics Research Corporation, Rapid Application Generator Development Specification, DS 620144502, November 1, 1985.

[12] Structural Dynamics Research Corporation, Text Editor Development Specification, DS 620144600B, November 1, 1985.

[13] Structural Dynamics Research Corporation, Application Interface Development Specification, DS 620144700, November 1, 1985.

[14] Structural Dynamics Research Corporation, Forms Language Compiler Development Specification, DS 620144401B, November 1, 1985.

[15] Structural Dynamics Research Corporation, Forms Driven Form Editor Development Specification, DS 620144402B, November 1, 1985.

[16] Structural Dynamics Research Corporation, Virtual Terminal Interface Development Specification, DS 620144300B, November 1, 1985.

[17] Structural Dynamics Research Corporation, User Interface Services Development Specification, DS 620144100B, November 1, 1985.

[18] Structural Dynamics Research Corporation, Form Processor Development Specification, DS 620144200B, November 1, 1985.

[19] "Programming Language Constructs for Screen Definition", L. A. Rowe and K. A. Shoens, IEEE Trans. on Software Eng. 9(1), 1983.

[20] THE PSYCHOLOGY OF HUMAN-COMPUTER INTERACTION, S. K. Card, T. P. Moran and A. Newell, Lawrence Erlbaum Associates, Inc. (1983).

[21] SOFTWARE PSYCHOLOGY: HUMAN FACTORS IN COMPUTER AND INFORMATION SYSTEMS, B. Shneiderman; Little, Brown and Co. (1982).

[21] General Electric Co., System Design Specification, 7 February 1983.

2.2 Terms and Abbreviations

American Standard Code for Information Interchange: (ASCII), the character set defined by ANSI X3.4 and used by most computer vendors.

Application Interface: (AI), subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The AI enables applications to be hosted on computers other than the host of the User Interface.

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Attribute: field characteristic such as blinking, highlighted, black, etc. and various other combinations. Background attributes are defined for forms or windows only. Foreground attributes are defined for items. Attributes may be permanent, i.e., they remain the same unless changed by the application program, or they may be temporary, i.e., they remain in effect until the window is redisplayed.

Device Drivers: (DD), software modules written to handle I/O for a specific kind of terminal. The modules map terminal specific commands and data to a neutral format. Device Drivers are part of the UI Virtual Terminal.

Display List: is similar to the open list, except that it contains only those forms that have been added to the screen and are currently displayed on the screen.

Extended Binary Coded Decimal Interchange Code: (EBCDIC), the character set used by a few computer vendors (notably IBM) instead of ASCII.

Field: two-dimensional space on a terminal screen.

Form: structured view which may be imposed on windows or other forms. A form is composed of fields. These fields may be defined as forms, items, and windows.

Form Definition: (FD), forms definition language after compilation. It is read at runtime by the Form Processor.

Forms Definition Language: (FDL), the language in which electronic forms are defined.

Forms Driven Form Editor: (FDFE), subset of the FE which consists of a forms driven application used to create Form Definition files interactively.

Form Editor: (FE), subset of the IISS User Interface that is used to create definitions of forms. The FE consists of the Forms Driven Form Editor and the Forms Language Compiler.

Form Hierarchy: a graphic representation of the way in which forms, items and windows are related to their parent form.

Forms Language Compiler: (FLAN), subset of the FE that consists of a batch process that accepts a series of forms definition language statements and produces form definition files as output.

Form Processor: (FP), subset of the IISS User Interface that consists of a set of callable execution time routines available to an application program for form processing.

Form Processor Text Editor: (FPTE), subset of the Form Processor that consists of software modules that provide text editing capabilities to all users of applications that use the Form Processor.

IISS Function Screen: the first screen that is displayed after logon. It allows the user to specify the function he wants to access and the device type and device name on which he is working.

Integrated Information Support System: (IISS), a test computing environment used to investigate, demonstrate and test the concepts of information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Item: non-decomposable area of a form in which hard-coded descriptive text may be placed and the only defined areas where user data may be input/output.

<u>Message</u>: descriptive text which may be returned in the standard message line on the terminal screen. They are used to warn of errors or provide other user information.

<u>Message Line</u>: a line on the terminal screen that is used to display messages.

<u>Network Transaction Manager</u>: (NTM), IISS subsystem that performs the coordination, communication and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

<u>Open List</u>: a list of all the forms that have been and are currently open for an application process.

<u>Operating System</u>: (OS), software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

<u>Page</u>: instance of forms in windows that are created whenever a form is added to a window.

<u>Paging and Scrolling</u>: a method which allows a form to contain more data than can be displayed with provisions for viewing any portion of the data buffer.

<u>Physical Device</u>: a hardware terminal.

<u>Qualified Name</u>: the name of a form, item or window preceded by the hierarchy path so that it is uniquely identified.

<u>Subform</u>: a form that is used within another form.

<u>User Data</u>: data which is either input by the user or output by the application programs to items.

<u>User Interface</u>: (UI), IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System (UIDS) and the User Interface Management System (UIMS).

<u>User Interface Development System</u>: (UIDS), collection of IISS User Interface subsystems that are used by applications

programmers as they develop IISS applications. The UIDS includes the Form Editor and the Application Generator.

User Interface Management System: (UIMS), the runtime UI. It consists of the Form Processor, Virtual Terminal, Application Interface, the User Interface Services and the Text Editor.

User Interface Monitor: (UIM), part of the Form Processor that handles messaging between the NTM and the UI. It also provides authorization checks and initiates applications.

User Interface Services: (UIS), subset of the IISS User Interface that consists of a package of routines that aid users in controlling their environment. It includes message management, change password, and application definition services.

User Interface/Virtual Terminal Interface: (UI/VTI), another name for the User Interface.

Virtual Terminal: (VT), subset of the IISS User Interface that performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by the UI software which constitutes the virtual terminal definition. Specific terminals are then mapped against the virtual terminal software by specific software modules written for each type of real terminal supported.

Window: dynamic area of a terminal screen on which predefined forms may be placed at run time.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 3

REQUIREMENTS

3.1 Computer Program Definition

The Forms Driven Form Editor provides an interactive tool for constructing and viewing forms. The tool provides the capability to store and select user forms.

3.1.1 Interface Requirements

The Forms Driven Form Editor (FDFE) interfaces directly with users as a UIS application. Physical terminals are assumed to have a video display, a textual keyboard, four cursor positioning keys or key sequences, a help key or key sequence, an entry key, and four other keys to be used by the FDFE for special processing. The FDFE must interface with the AI, FLAN and the operating system.

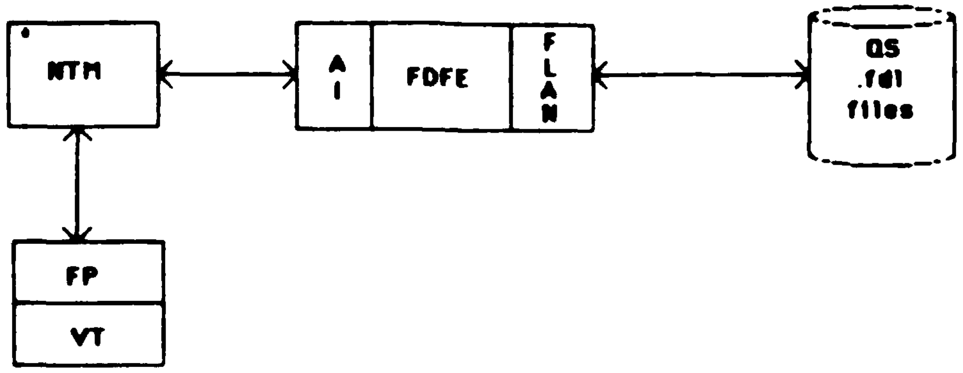


Figure 3-1 Interface Diagram

3.1.1.1 Application Interface

The FDFE interacts with users by calling appropriate routines of the Application Interface (AI). This interface creates messages which are sent to the Form Processor which moves information describing interactive terminal input and output and provides a link to users of the FDFE through the Virtual Terminal.

3.1.1.2 Stored Forms Management Interface

The FDFE uses the Forms Language Complier (FLAN) to convert forms language source into the Form Processor internal forms structure. The FDFE also invokes the FLAN when the form under construction is to be stored.

3.1.1.3 Operating System Interface

The FDFE stores form source language files (*fdl files on the VAX) and compiled form definitions (*fd files on the VAX). Forms language files may subsequently be compiled and displayed. The storage of the fdl files and fd files is system dependent. The VAX implementation uses the logicals IISSSLIB (for fdl files) to store/retrieve the files in/from the appropriate directory.

3.1.1.4 User Interface Displays

The layout of the screens discussed in the following section may be found in Appendix A.

The user interface provides WORK TASKS for constructing and modifying form language sources and compiled form definitions:

1) List Form Language Sources - list all form language sources

2) Copy Form Language Source - copy from a form language source to a new form language source

3) Rename Form Language Source - Rename old form language source name to a new form language source name

4) Insert, Drop, Modify, Select Form Language Source - operations on the form language source

5) List Compiled Form Definitions - list all compiled form definitions

6) Display Compiled Form Definition - display the form as a user would view the form

7) Drop Compiled Form Definition - drop the compiled form definition

The user may choose any of these menu operations by marking the appropriate selection and filling in the necessary parameters for the operation. Also available to the user is command entry capability. The user types the appropriate command (the capital letters of the menu operation indicate the command) with its needed parameters in the order specified by the menu operation. Parameters are separated by blanks.

For each selected WORK TASK, the appropriate screen is displayed to complete the task or lead the user to the next task to be completed.

For the List tasks of the WORK TASK, the next screen displayed is the list of form language sources or compiled form definitions.

For the Copy, Rename and Drop tasks, the screen displayed next is the same screen with a success/error message displayed. For the Display task, the next screen displayed is the form itself with the appropriate instructional message on how to proceed.

For the Insert, Modify and Select tasks, the next screen displayed is the EDIT TASKS screen which provides the following operations:

1) List Forms - lists all forms created in the current form language source

2) Write Forms - saves the form language source

3) Write and Compile Forms - save both forms language source and compiled forms definition

4) Exit Compile forms - saves form language source and compiled form definitions for all forms created in the form language source and returns to work task s[illegible]

5) Insert, Modify and Select Form - operations that are performed on a form which then requires an edit mode selection

6) Drop Form - drop the specified form from the form language source

For the List Forms task, the next screen displayed lists all the forms created by the current form language source. For the Drop, Write Forms and Write and Compile Forms tasks, the next screen returned is the EDIT TASK Screen which contains the appropriate success/error messages.

For the Insert, Modify and Select Form tasks, the next screen displayed depends on the edit mode chosen:

1) Single Field - the edit operations done on the form are done on a per field basis

2) Form - the edit operations done on the form may be done on all the fields at once

3) Layout - for positioning and constructing fields of a form

Each mode contains several activities which facilitate use of that particular mode. The user is provided with visual feedback describing the mode/task combination in use at any particular time. The Work Tasks, Edit Tasks and Edit modes are described in detail in later sections.

3.1.1.5 Data Entry

The FDFE uses four types of data entry:

1) positional selection
2) textual selection
3) by-example
4) completion

Positional selection requires the user to position the cursor in specific locations and mark the selected activity. Textual selection provides a menu of items and requires entry of an abbreviated code to select an item. By-example allows entry of information as it will appear in the completed form. Completion data entry requires entry of text to complete a displayed form.

On the WORK TASKS, EDIT TASKS, FIELD EDIT, FORM EDIT and the LAYOUT DESCRIPTION screens, all but the by-example method of data entry is used. The user must choose, however, on the WORK TASK and EDIT TASK screens whether to use positional selection (using the menu specified operations) or textual selection (entering the abbreviated command, the capital letters of the descriptive text of the menu specified operations) in the command entry line. The LAYOUT EDIT screen is the only screen which allows data entry by example.

3.1.1.6 Function Keys

The only defined function keys for the Forms Driven Form Editor are a function key which, when hit, takes the user back to the previous logical screen, a function key to be used in LAYOUT EDIT MODE for getting to layout description editing, two function keys for going forwards and backwards through fields on a form, and a function key used in Layout Edit mode to move fields around on a form.

If the user returns to the previous logical screen without completing entry of the data on the current screen, this data is lost. In LAYOUT EDIT MODE, the user positions the cursor at the field on the screen that is to be further created and presses the "LAYOUT DESCRIPTION" function key. The resulting screen is the LAYOUT DESCRIPTION screen which the user may then complete.

The "Enter" key always implies continuing with the next logical step in the completion of the current task. The "Help" function key is used as defined by the Form Processor. Whenever the "Help" function key is pressed with the cursor positioned at an item, the help message or help form for that item is displayed. The help forms or messages for further describing items have not been presented but follow a consistent form so that the user is given complete information about the item for which help has been requested.

3.2 Detailed Function Requirements

The general approach is to view the FDFE as a hierarchy of modules. The FDFE screens presented in Appendix A are associated with only certain modules in the hierarchy based on the functionality being performed by the module.

3.2.1 Module Hierarchy

The following hierarchy chart shows the organization of the FDFE:

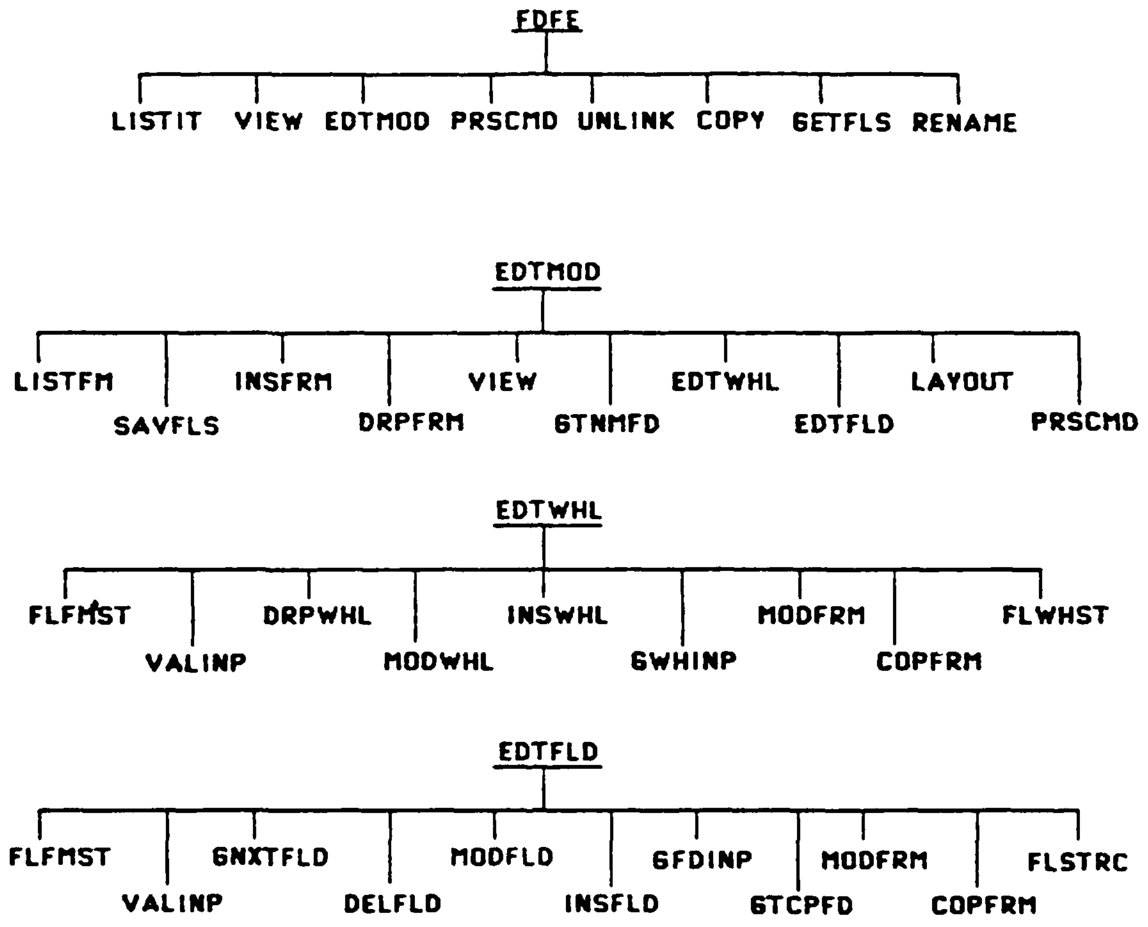


Figure 3-2a FDFE Hierarchy - Part 1

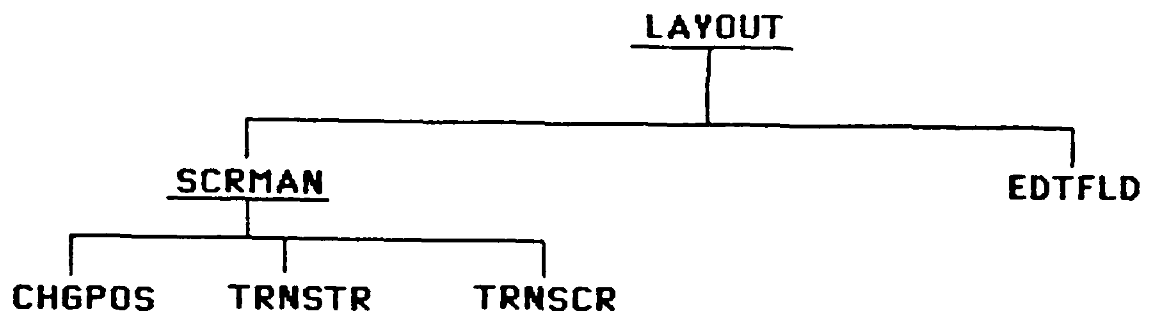


Figure 3-2b FDFE Hierarchy - Part 2

3.2.2 Module Descriptions

The following paragraphs describe the modules associated with each of the major sections of the FDFE.

MODULE DEFINITIONS

FLS = Forms Language Source
FDO = Forms Definition Object

3.2.2.1 FDFE

This module is the main driver. It allows the user to choose among several file management options or to proceed to the edit task level. It controls the WRKTASK screen.

Input Parameters:
 None
Output Parameters:
 None

3.2.2.2 PRSCMD

This module parses the command line for both the WRKTASK and EDTTASK screens to determine which other modules are to be called and what parameters are to be passed.

Input Parameters:
 Pointer to command line
 Array of parameter counts
 Array of pointers to valid commands

Output Parameters:
Option chosen
Pointer to parameters to be passed to the module which will execute the option.
Number of parameters found in command line

3.2.2.3 LISTIT

This module lists all of the Forms Definition Object or Forms Language Source in the user's specified forms language source or definition object libraries.

Input Parameters:
Pointer to "FDL" or "FD" string
Output Parameters:
Returns any error code or a NULL pointer if successful

3.2.2.4 VIEW

This module displays a form just as it would appear on the screen when used by a program.

Input Parameters:
Pointer to name of form to view
Output Parameters:
Returns any error code or a NULL pointer if successful

3.2.2.5 FORMS LANGUAGE SOURCE ACCESS MODULES

- UNLINK:

  This module drops a particular Forms Language Source.

  Input Parameters:
  Name of Forms Language Source
  Output Parameters:
  Returns any error code or a NULL pointer if successful

- COPY:

  This module copies a particular Forms Language Source to another Forms Language Source of specified name.

  Input Parameters:
  Name of existing "from" Forms Language Source
  Name of new or "to" Forms Language Source

Output Parameters:
  Returns any error code or a NULL pointer if successful

- RENAME:

  This module renames a particular Forms Language Source to a specified name.

  Input Parameters:
    Name of existing Forms Language Source
    New name of Forms Language Source
  Output Parameters:
    Returns any error code or a NULL pointer if successful

- GETFLS:

  This module retrieves a particular Forms Language Source and translates it into the internal data structure.

  Input Parameters:
    Name of existing Forms Language Source
  Output Parameters:
    Pointer to the "opened" form, the internal Forms Processor data structure

- SAVFLS:

  This module saves a particular Forms Language Source (fdl file) and a Forms Definition Object (fd file) after it translates the internal data structure into forms language syntax.

  Input Parameters:
    Name of the form to be saved under flag indicating to write or not to write out the fd file
    Pointer to list of forms to be written out
  Output Parameters:
    Returns any error code or a NULL pointer if successful

3.2.2.6 EDTMOD

This module is the control module for all edit tasks. It controls the EDTTASK screen.

Input Parameters:
  New/old form flag
  Change/retrieve only flag

Output Parameters:
  Returns any error code or a NULL pointer if successful

3.2.2.7 LISTFM

This module lists all forms in the Forms Language Source on which work is being done.

Input Parameters:
  None
Output Parameters:
  Returns any error codes or NULL pointer if successful

3.2.2.8 INSFRM

This module inserts a new form into the Forms Language Source on which work is being done.

Input Parameters:
  name of form
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.9 DRPFRM

This module deletes a form from the Forms Language Source on which work is being done.

Input Parameters:
  Name of form
Output Parameters:
  Returns any error codes or NULL pointer if successful

3.2.2.10 EDTWHL

This module allows the user to edit an entire form at once. It controls the presentation of the FORM EDIT screen.

Input Parameters:
  Read-only flag
  Name of form
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.11 EDTFLD

This module allows the user to edit all fields of a form one at a time. It controls the presentation of the FIELD EDIT screen. It is also called from LAYOUT.

Input Parameters:
  Read-only flag
  col cursor position if coming from layout mode
  row cursor position if coming from layout mode
  pointer to internal form structure
  edit mode
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.12 LAYOUT

This module allows the user to edit an entire form as it would appear when used (with regards to the location and size of fields) on one screen. The other needed information is filled in on the LAYOUT DESCRIPTION screen.

Input Parameters:
  Pointer to internal form structure
  Read-only fla
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.13 SCRMAN

This module controls the first stage of layout edit mode - it manages the screen using the following three modules to translate internal structure to screen layout and vice versa.

Input Parameters:
  Read-only flag
  pointer to internal form structure
Output Parameters:
  Row position returned from GETCUR
  Col position returned from GETCUR

3.2.2.14 CHGPOS

This module allows the user to change the location of a field in layout mode by indicating the "from" and "to" locations on the screen.

Input Parameters:
  pointer to internal form structure
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.15 TRNSCR

This module translates the layout screen format to internal structure.

Input Parameters:
  Pointer to internal form structure
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.16 TRNSTR

This module translates the internal structure to the layout screen format.

Input Parameters:
  Pointer to internal form structure
  Read only flag
Output Parameters:
  None

3.2.2.17 VALINP

This module performs validation checks on fields. The objects to be validated are the values input on the FIELD EDIT and FORM EDIT screens.

Input Parameters:
  Pointer to form to be validated
  Pointer to field to be validated
  Flag indicating type of validation
Output Parameters:
  Returns TRUE if validation okay else returns FALSE

3.2.2.18 GTNMFD

This module retrieves fields from the internal structure.

Input Parameters:
  Pointer to 1st field in internal structure
  Name of field to find
Output Parameters:
  Pointer to field in the internal structure or NULL if could not find field

3.2.2.19 MODFLD

This module modifies a field in the internal structure.

Input Parameters:
  Pointer to parent of field
  Pointer to pointer of field being modified
  Pointer to screen changed information
  Pointer to screen help info
  Pointer to screen value info
  Pointer to screen item info
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.20 DELFLD

This module deletes a field from the internal structure. This is the same function as that used by the Form Processor.

3.2.2.21 INSFLD

This module inserts a field into the internal structure.

Input Parameters:
  Address of pointer to field being inserted
  Address where next field pointer will be inserted
  Address where previous field pointer will be inserted
  Pointer to parent of field
  Pointer to screen field information
  Pointer to screen help info
  Pointer to screen value info
  Pointer to screen item info
  Recursion level

Output Parameters:
  Returns any error code or NULL pointer if sucoessful

3.2.2.22 COPFRM

This module copies a Forms Language Source file into an alternate internal data structure and gets the pointer to the specified form.

Input Parameters:
  Name of Forms Language Source file to copy from
  Name of form to copy
Output Parameters:
  Sets global variables:
  copyfls, name of Forms Language Source just copied
  Copyfrm, name of form just copied
  Altbuf, beginning of list containing all forms of Forms
  Language Source
  Altfrm, pointer to form user wishes to copy

3.2.2.23 MODFRM

This module updates information about the form.

Input Parameters:
  Pointer to form internal structure
  Pointer to screen form information
Output Parameters:
  Returns any error code or NULL pointer if successful

3 2 2 24 FLFMST

This module translates an FPD field structure for a form into the screen information structure.

Input Parameters
  Pointer to screen structure
  Pointer to fpd form field
Output Parameters
  None

3 2 2 25 FLSTRC

This module translates the FPD field structure to screen information structure for items, windows, and forms.

Input Parameters:
  Pointer to fpd field
  Pointer to screen field structure
  Pointer to screen item help structure
  Pointer to screen item value structure
  Pointer to screen field domain check structure
Output Parameters:
  Fills appropriate screen area with field information

3.2.2.26 FLWHST

This module fills in the output screen FORM EDIT, and associates each field line on the output screen with the field's internal structure.

Input Parameters:
  Pointer to form internal structure on which editing is to occur.
Output Parameters:
  Fills output screen FORM EDIT area with form and field info and creates an external array of pointers

3.2.2.27 GWHINP

This module gets all input for the FORM EDIT screen for the fields on the form being edited.

Input Parameters:
  Pointer to form internal structure
Output Parameters:
  PF key provided by OISCR
  Returns any error code or NULL pointer if successful

3.2.2.28 GTCPFD

This module gets the field at the located cursor position.

Input Parameters:
  Pointer to form internal structure
  Row cursor position
  Col cursor position
Output Parameters:
  Pointer to field at that location else NULL if no field found

3.2.2.29 DRPWHL

This module deletes all fields marked by the user on the FORM EDIT screen.

Input Parameters:
  Pointer to internal form structure
Output Parameters:
  PF key received by OISCR

3.2.2.30 MODWHL

This module modifies existing fields as input by the user on the FORM EDIT screen.

Input Parameters:
  Pointer to parent of field
  Pointer to field being modified
  Pointer to input screen structure
  Pointer to help line on screen
  Pointer to item value on screen
  Pointer to item only info on screen
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.2.31 INSWHL

This module inserts all fields that have been entered on the FORM EDIT screen.

Input Parameters:
  Pointer to form internal structure
Output Parameters:
  Returns any error code or NULL pointer if successful

3.2.3 Edit Modes

The following sections describe the modes available within the FDFE EDIT TASKS of Insert, Modify, and Select Form. Appendix A contains a layout of the associated screens that are used in each mode.

3.2.3.1 Field Edit Mode

The Field Edit Mode displays the form which describes the target form being constructed. The user enters textual information to complete information about the current form. The

Field Edit Mode allows the user to individually drop, modify, select and insert fields within the current form. It collects the textual information entered by the user for purposes of modifying and retrieving a field. In addition, the user may select a field from a form other than the current form by completing the textual information of the name of the copy form language source and the copy form within the copy form language source. The user may navigate through all fields on the current form by using the select operation and specifying the field type and the direction, next or previous.

3.2.3.2 Layout Edit Mode

Layout Edit Mode provides "by-example" capabilities for defining forms. These capabilities can be divided into two categories. One category provides the ability to visually position subforms, items, and windows, to associate prompts with fields, and to visually move fields on the form. This capability saves form design time since positional parameters and field sizes are calculated for the user by Layout Edit Mode.

The second capability provides a convenient way to complete development of a form by using the Field Edit format to describe the necessary information about the fields positionally inserted in Layout Edit Mode. The user positions the cursor on the field to be further described and uses the Layout Description key to select the next form. This form coaches the user on the necessary information to fully insert a field. This form also provides copy capability which allows the user to select a field from another form and position it in the current work form.

The user is not required to go into the Layout Description form for the positionally inserted fields since the FDFE makes default assumptions about the required information for field definition. The following defaults are assumed: the field type is an item and the display attribute is INPUT. Field names are generated based on the row and column location of the field.

3.2.3.3 Form Edit Mode

The Form Edit Mode provides the user with a quick way of entering all the fields associated with a form at one time. The user may insert, modify, select, and drop multiple fields using this one form. The initial subform placed in the variable window allows the user to insert all the necessary information for any type of field: subform, window or item. The user may then further continue definition of item fields by marking the

More Item Description operation contained within the subform. The user continues to do so until the item information is completed.

3.2.4 Errors Management

The FDFE integrates with the Form Processor to handle three types of errors:

1) input syntax
2) input content
3) disruptive

Input syntax errors and errors in content values for items are handled within the module processing the user data coming in from its associated screens. Disruptive errors such as running out of memory or program bugs in Form Processor calls, can be caused by system activities and are handled by calling the standard system message routine which displays the message to the user on the current screen.

The FDFE modules use the Form Processor to provide proper error code mapping and, where possible, use existing FP error handling.

3.3 Performance Requirements

A performance goal is to achieve a one second response time following user input. This goal has been verified in several research efforts, but such a short time can rarely be achieved in large, centralized systems under realistic loads. Multi-processing systems which use microcomputers for terminals, with significant downloading, have been able to satisfy this goal.

A backup position for system performance is to provide a predictable response time for users. Users have been found to accept delays of several seconds if they occur in a repeatable manner. Users also will tolerate longer response times in situations where they perceive that the activity is complex.

3.3.1 Program Organization

The FDFE is organized such that access to data whether it be within the CDM or from a file system is only through calling the appropriate access routine. Also, not all modules are responsible for handling processing of the screens. In addition, a controlling model is responsible for forms handling for each level of the module hierarchy.

3.4 Human Performance

The following sections discuss several goals related to human performance considerations.

3.4.1 Minimization of Keypad Overloading

Each key should be used for only a small set of functions. Unambiguous input operations improve editor operation by reducing confusion. Positioning devices such as mice have been shown to be superior to keypads.

3.4.2 Overviewing

Complex forms may require representation on many screens. A method of presenting an abbreviated view of an entire parent form on one screen allows the user to navigate through the form by selecting subforms, windows, etc.

3.4.3 Graphical Representations and Icons

These approaches provide more information in limited screen landscapes and support a more powerful editor environment.

3.4.4 Undo Functions

Undo capability for previous actions facilitates exploration of editor functions for novices, and also allows experienced users to work at a more rapid pace. This feature was not implemented per se but mistakes may be corrected through careful use of the provided functions of the FDFE.

3.5 Data Base Requirements

3.5.1 Sources and Types of Input

The major source of input comes from processing of the user data entered on the FDFE forms. This input is error-checked and then translated into an internal structure. The internal structure is that used by the form processor. Another source of input is the form language source of the FDFE user.

3.5.2 Destinations and Types of Output

The output of the FDFE is the translation of the internal structure into the forms language source and the compiled form definition objects. This translation occurs at the point of saving the edited changes.

3.5.3 Internal Tables and Parameters

Internal tables consist of valid commands at the Work Task and Edit Task levels along with the number of required parameters for these commands.

3.6 Help Forms

Input items have an associated help message or help form which may be retrieved by using the standard Form Processor "help" key.

SECTION 4

QUALITY ASSURANCE PROVISIONS

4.1 Introduction and Definition

"Testing" is a systematic process that may be preplanned and explicitly stated. Test techniques and procedures may be inserted in advance and a sequence of test steps may be specified. "Debugging" is the process of isolation and correction of the cause of an error.

4.2 Computer Programming Test and Evaluation

The quality assurance provisions for test consist of the normal testing techniques that are used during the construction process. They consist of design and code walk-throughs, unit testing, and integration testing. These tests are performed by the design team.

Each function is tested separately, then the entire sub-system is tested as a unit. All testing except for integration with software belonging to other companies is done at SDRC on the VAX.

The integration testing entails use of the scripting capability which executes major functions of the FDFE. The script file and its results may be used to verify the FDFE after release.

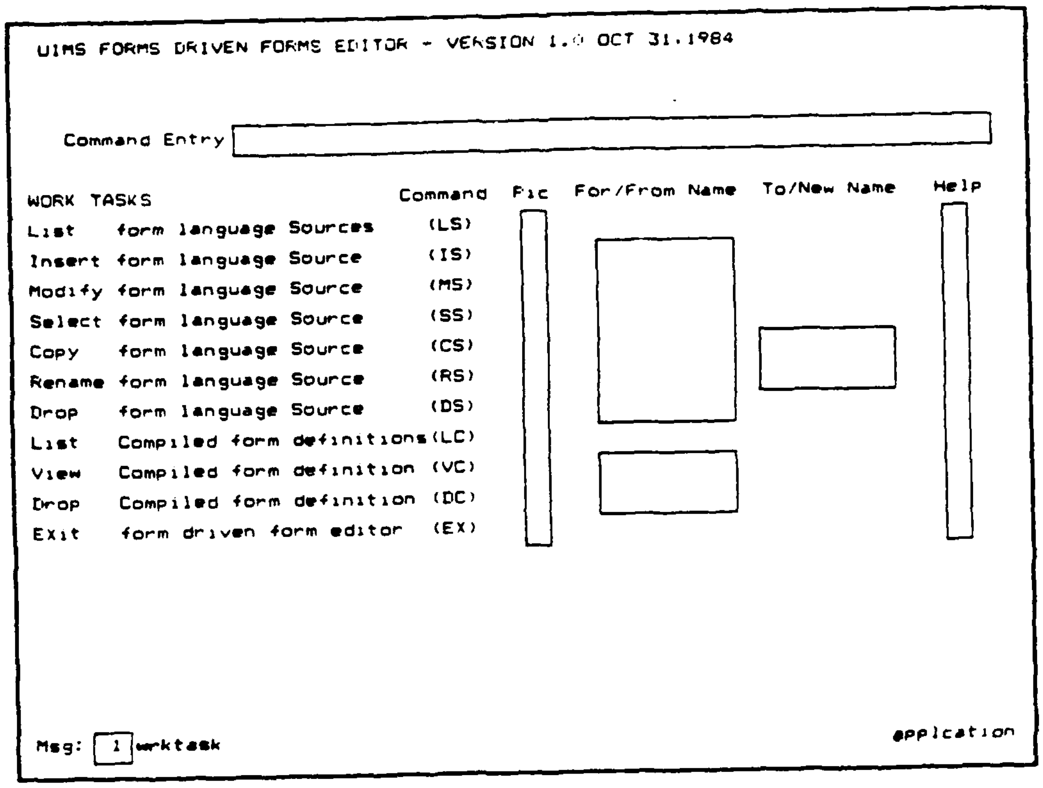
SECTION 5

PREPARATION FOR DELIVERY

The implementation site for the constructed software is the ICAM Integrated Support System (IISS) Test Bed site located at General Electric, Schenectady, New York. The software associated with each FDFE release is delivered on a medium which is compatible with the IISS Test Bed. The release is clearly identified and includes instructions on procedures to be followed for installation of the release.

APPENDIX A

FORMS DRIVEN FORM EDITOR SCREENS



UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984

Command Entry

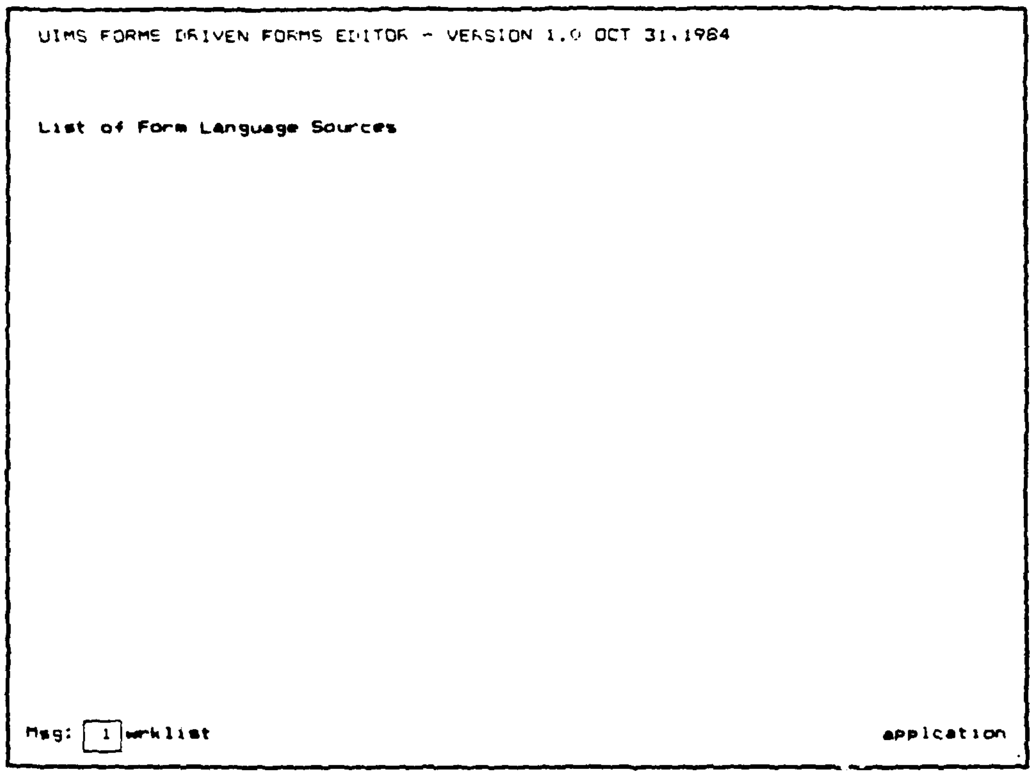
WORK TASKS	Command
List form language Sources	(LS)
Insert form language Source	(IS)
Modify form language Source	(MS)
Select form language Source	(SS)
Copy form language Source	(CS)
Rename form language Source	(RS)
Drop form language Source	(DS)
List Compiled form definitions	(LC)
View Compiled form definition	(VC)
Drop Compiled form definition	(DC)
Exit form driven form editor	(EX)

Pic For/From Name To/New Name Help

Msg: 1 wrktask

applcation

Figure A-1 Screen 1



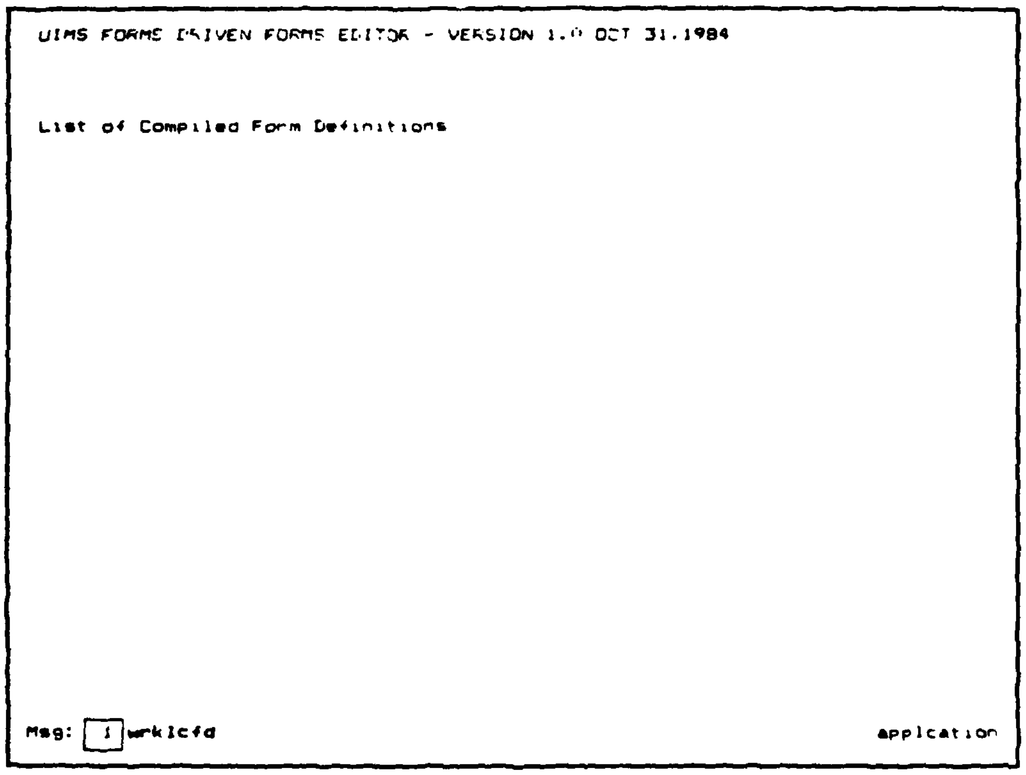
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984

List of Form Language Sources

Msg: 1 wrklist

applcation

Figure A-2 Screen 2

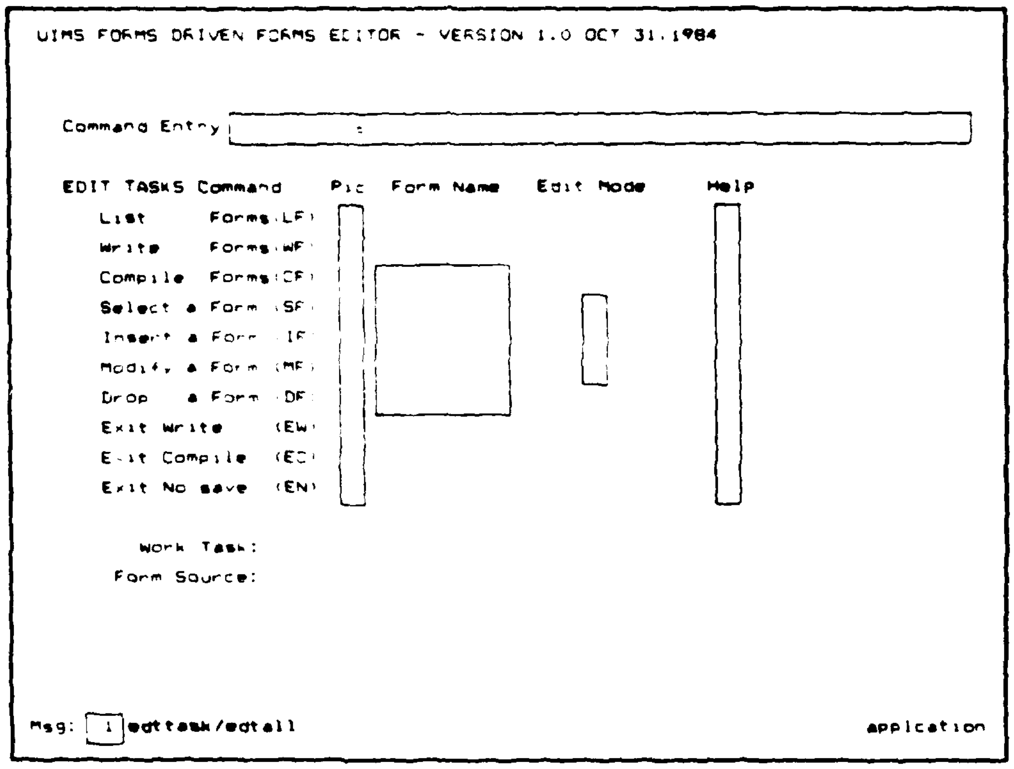

UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31.1984

List of Compiled Form Definitions

Msg: 1 wrklcfd

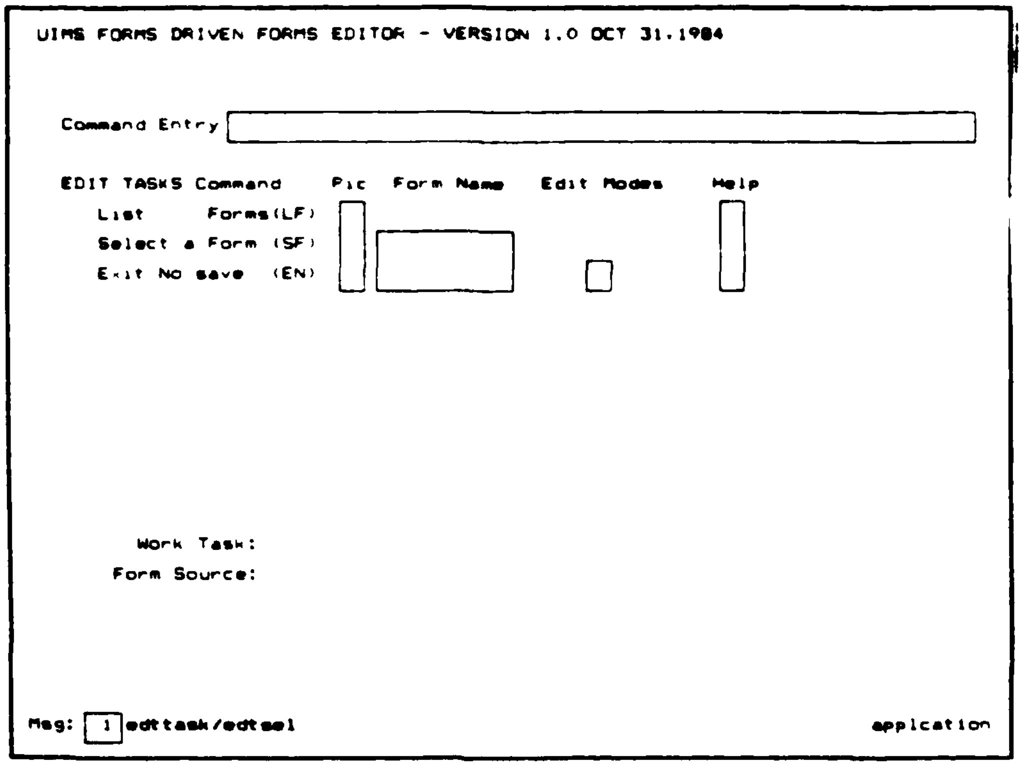
applcation

Figure A-3 Screen 3



```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984

Command Entry

EDIT TASKS Command     Pic  Form Name  Edit Mode   Help
   List      Forms(LF)
   Write     Forms(WF)
   Compile   Forms(CF)
   Select  a Form (SF)
   Insert  a Form (IF)
   Modify  a Form (MF)
   Drop    a Form (DF)
   Exit Write     (EW)
   Exit Compile   (EC)
   Exit No save   (EN)

     Work Task:
   Form Source:

Msg: 1 edttask/edtall                                application
```

Figure A-4 Screen 4



UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31.1984

Command Entry

EDIT TASKS Command   Pic   Form Name   Edit Modes   Help

List Forms (LF)

Select a Form (SF)

Exit No save (EN)

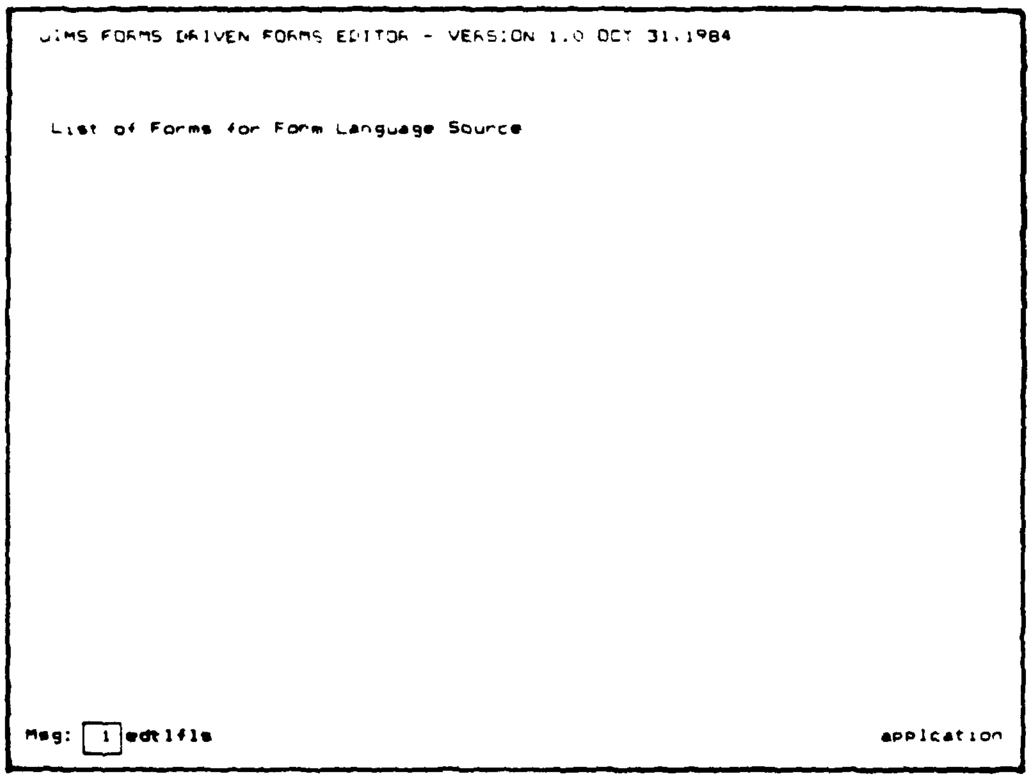
Work Task:

Form Source:

Msg: 1 edttask/edtsel

application

Figure A-5 Screen 5



```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984


 List of Forms for Form Language Source
















Msg: [1]edtlfls                                                   application
```

Figure A-6 Screen 6

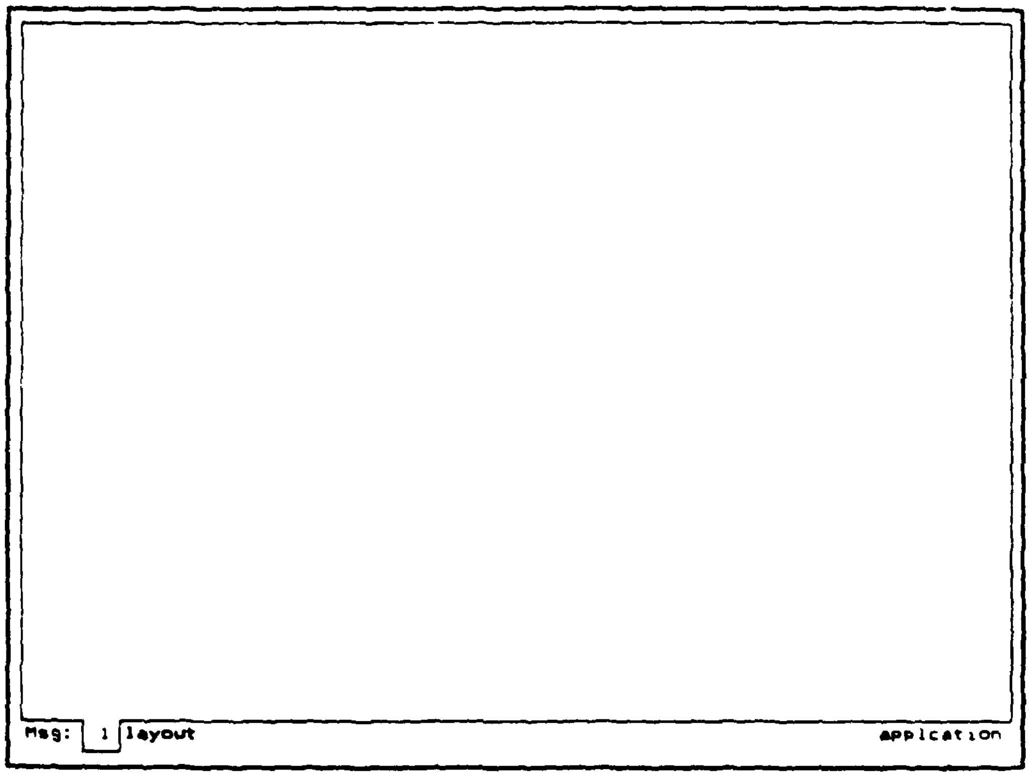
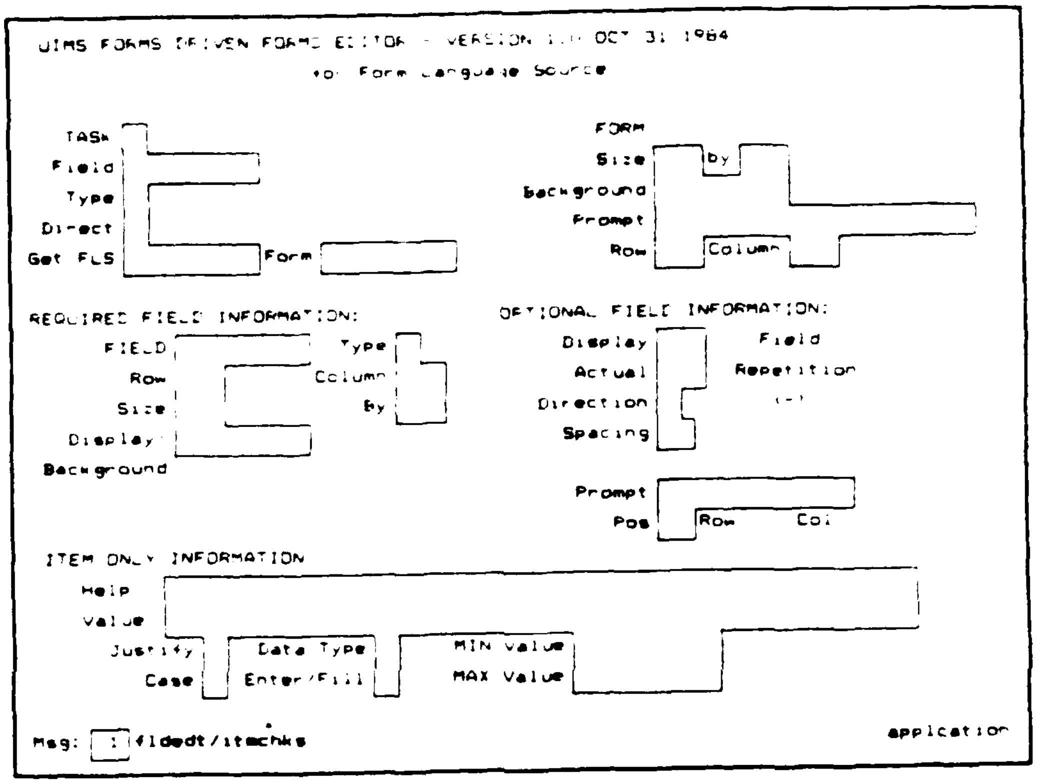
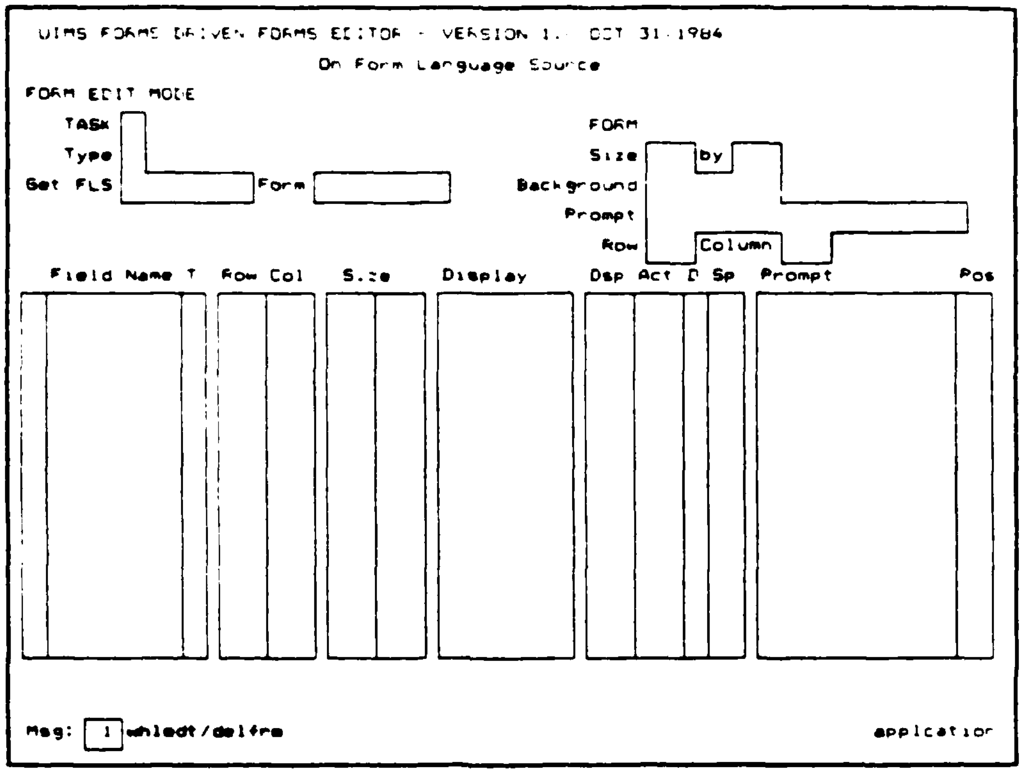


Figure A-7 Screen 7



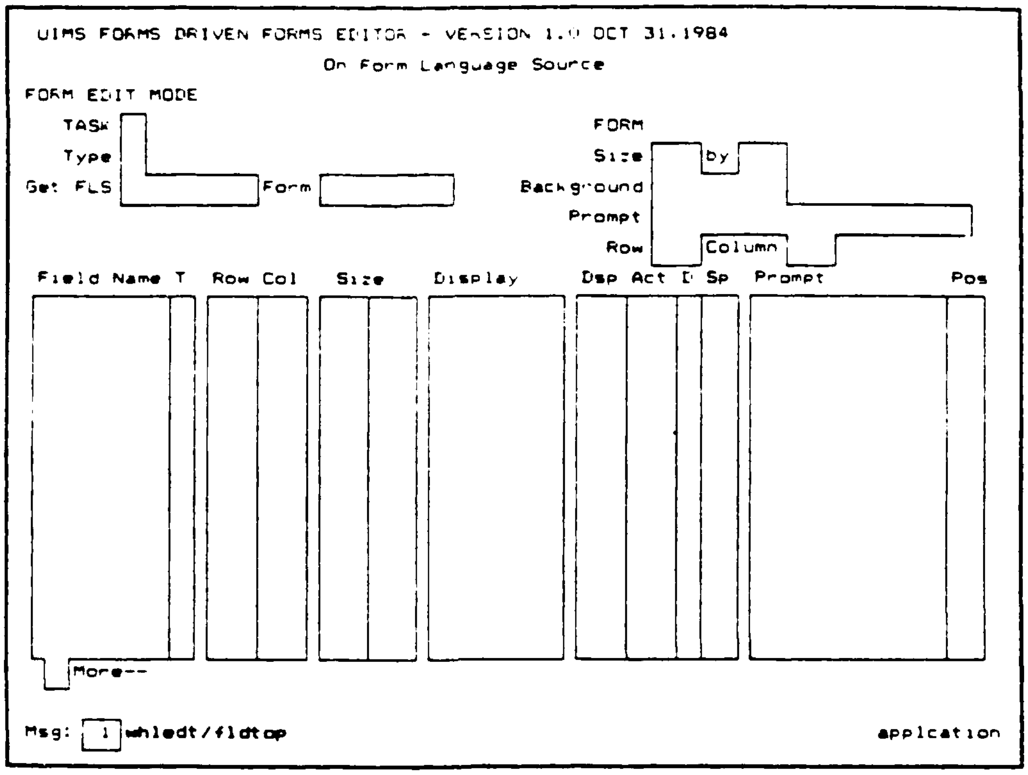
```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31 1984
                 for Form Language Source

     TASK                                   FORM
    Field                                   Size      by
     Type                             Background
   Direct                                 Prompt
  Get FLS          Form                      Row      Column

REQUIRED FIELD INFORMATION:          OPTIONAL FIELD INFORMATION:
        FIELD            Type               Display        Field
          Row            Column              Actual        Repetition
         Size            By               Direction        (-)
      Display                               Spacing
   Background
                                             Prompt
                                                Pos   Row       Col
 ITEM ONLY INFORMATION
     Help
    Value
      Justify     Data Type        MIN Value
         Case     Enter/Fill       MAX Value

Msg: 1 fldedt/itmchks                                    application
```

Figure A-8 Screen 8



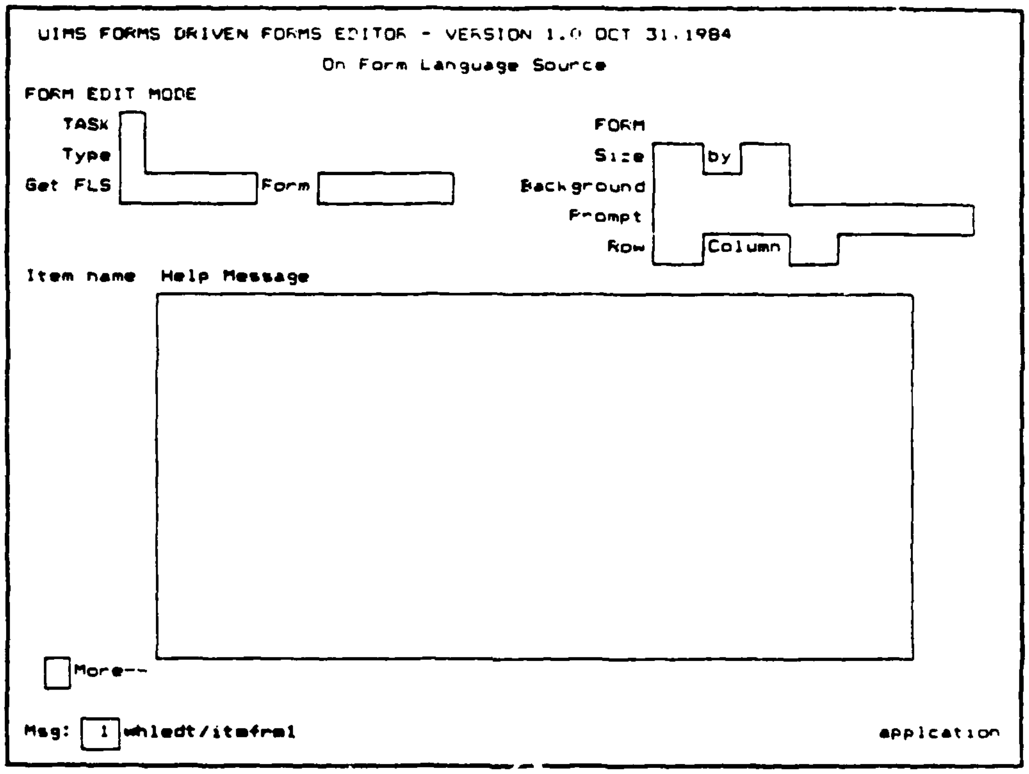
```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.  OCT 31 1984
                 On Form Language Source
FORM EDIT MODE
     TASK                                    FORM
     Type                                    Size      by
 Get FLS              Form             Background
                                           Prompt
                                              Row      Column
  Field Name T  Row Col  Size  Display   Dsp Act D Sp Prompt        Pos

Msg: 1 whledt/delfrm                                      applcation
```

Figure A-9 Screen 9



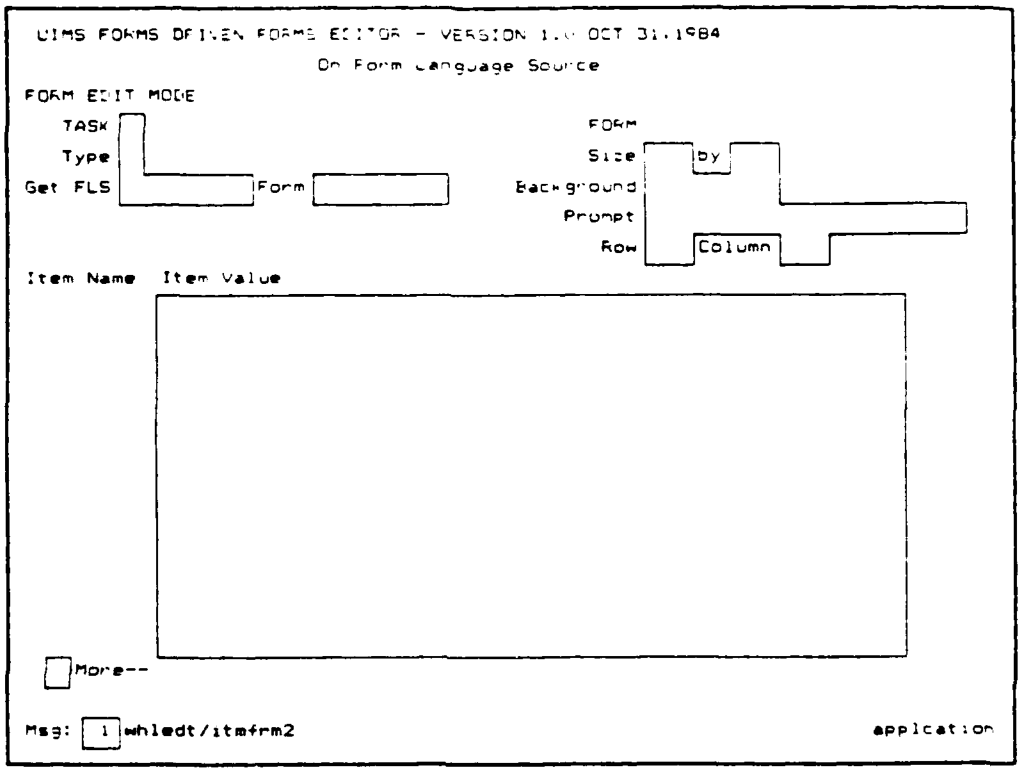
```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31.1984
                    On Form Language Source
FORM EDIT MODE
    TASK                                        FORM
    Type                                        Size       by
Get FLS                  Form             Background
                                              Prompt
                                                 Row       Column
Field Name T  Row Col  Size  Display  Dsp Act I Sp Prompt       Pos

 More--

Msg: 1 whledt/fldtop                                    applcation
```

Figure A-10 Screen 10



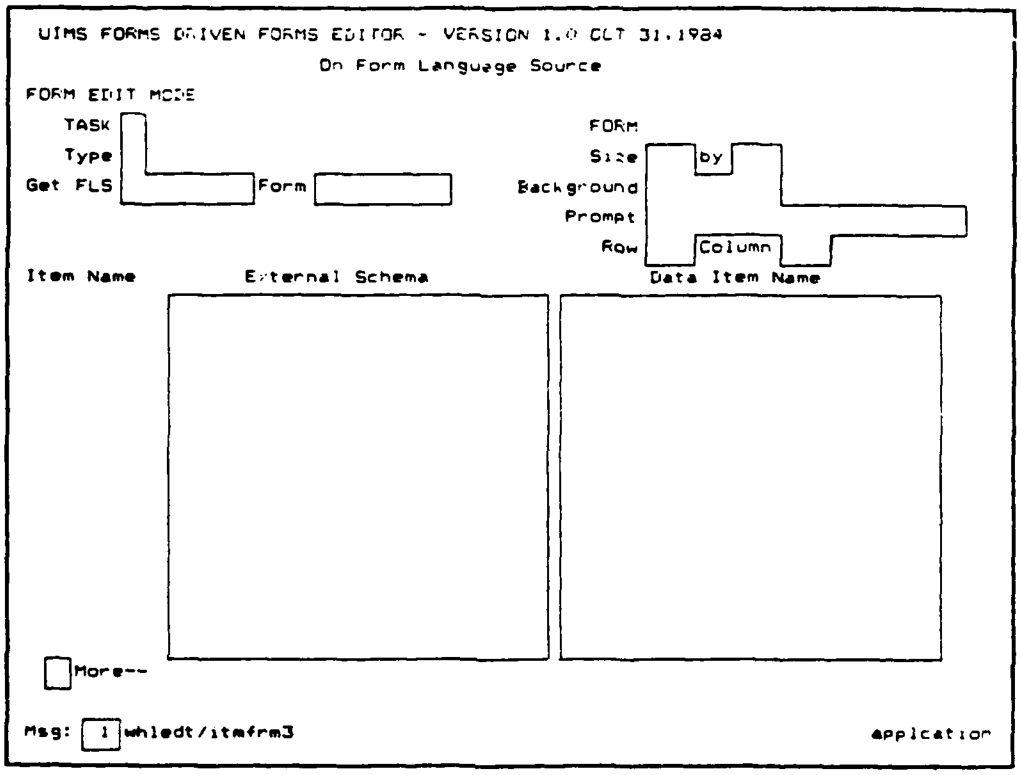
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984
On Form Language Source
FORM EDIT MODE
TASK
Type
Get FLS
Form
FORM
Size
by
Background
Prompt
Row
Column
Item name
Help Message
More--
Msg: 1 whledt/itmfrml
application

Figure A-11 Screen 11



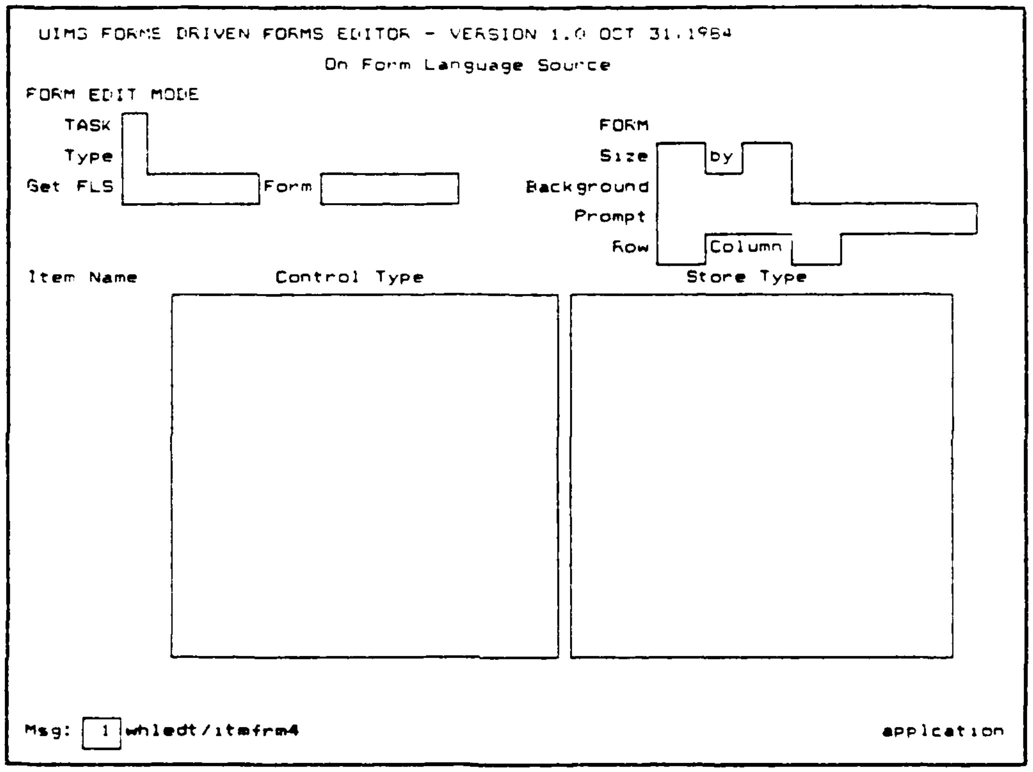
```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984
                 On Form Language Source
FORM EDIT MODE
   TASK                                      FORM
   Type                                      Size       by
Get FLS              Form              Background
                                         Prompt
                                            Row      Column
Item Name  Item Value

 More--

Msg: 1 whledt/itmfrm2                                  applcation
```

Figure A-12 Screen 12



```
UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984
                         On Form Language Source
FORM EDIT MODE
   TASK                                         FORM
   Type                                         Size        by
Get FLS               Form                  Background
                                                Prompt
                                                   Row       Column
Item Name          External Schema               Data Item Name

 More--

Msg: 1 whledt/itmfrm3                                        application
```

Figure A-13 Screen 13



UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984

On Form Language Source

FORM EDIT MODE

TASK

Type

Get FLS | Form

FORM

Size | by

Background

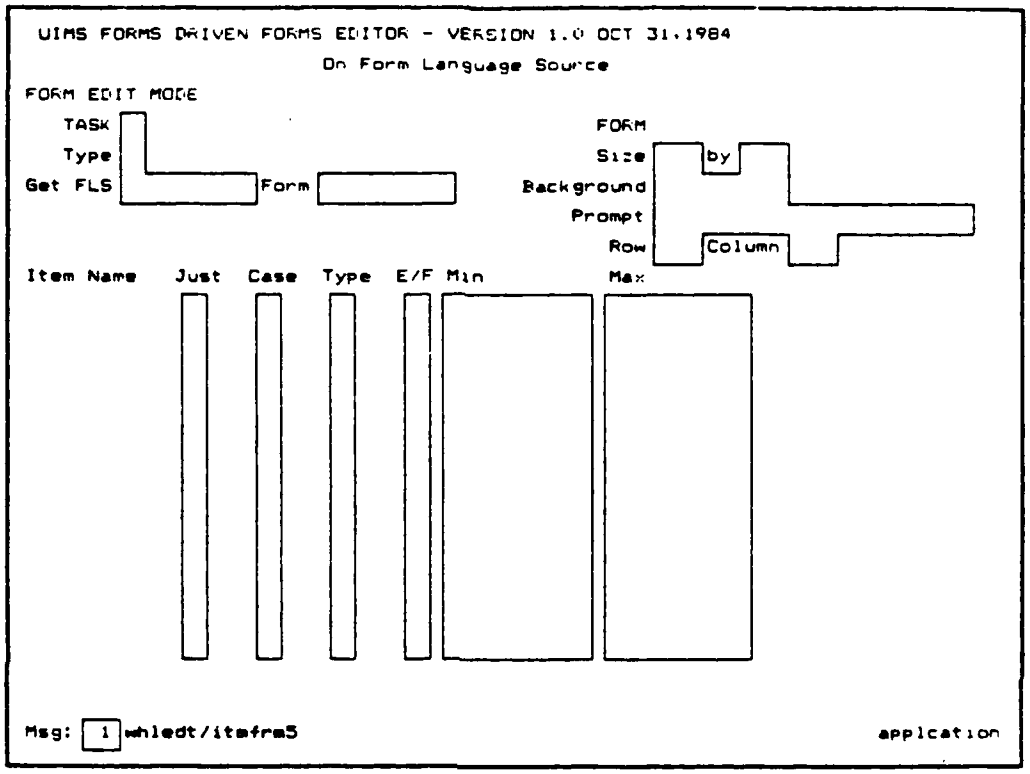
Prompt

Row | Column

Item Name | Control Type | Store Type

Msg: 1 whledt/itmfrm4 | applcation

Figure A-14 Screen 14



UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31,1984
On Form Language Source

FORM EDIT MODE

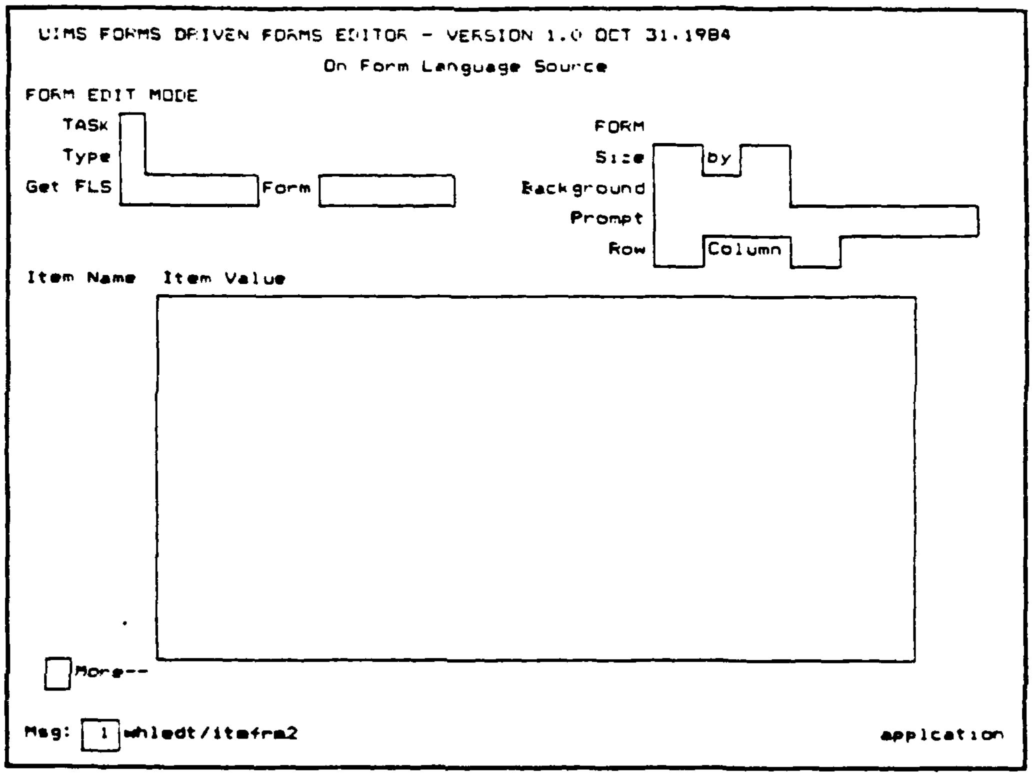
TASK
Type
Get FLS   Form

FORM
Size   by
Background
Prompt
Row   Column

Item Name   Just   Case   Type   E/F   Min   Max

Msg: 1 whledt/itmfrm5   applcation

Figure A-15 Screen 15



UIMS FORMS DRIVEN FORMS EDITOR - VERSION 1.0 OCT 31.1984

On Form Language Source

FORM EDIT MODE

TASK

Type

Get FLS

Form

FORM

Size by

Background

Prompt

Row Column

Item Name Item Value

More--

Msg: 1 whledt/itmfrm2

applcation

Figure A-16 Screen 12

END
8-87
DTIC